Radial Basis Function (RBF) Neural Networks Based on the Triple Modular Redundancy Technology (TMR)

Yaobin Qin qinxx143@umn.edu
Supervisor: Pro.lilja
Department of Electrical and Computer Engineering

Abstract

Neural networks are family statistical learning algorithms and structures and are used to estimate or approximate functions and pattern classification. The Neural network system is constructed through interconnected neurons and training weights. The paper will present the improvement of recognition rate, recognition time and hardware overhead through introducing TMR technology into the conventional RBF neural network which is a simple neural network only consisting of three layers.

Introduction

Due to the non-linear approximation of Radial Basis Functions Neural network, it is employed for functional approximation in time-series modeling and in pattern classification, specifically, such as potential function, clustering, functional approximation, spline interpolation and so on [1]. RBF neural network is a simple neural network consisting of 3 layers, input, hidden and output layer. Input layer is used to input the features of the pattern. In the hidden layer, each interconnection between input and hidden units implements the radial activated function. And the selection of activated function depends on the specific application. Output layer implements a linear combination of the hidden unit outputs to identify the specific pattern.

Given a RBF neural network, we need to specify the activated functions, the number of the neurons, the rules for modeling a given task and finding the characteristics parameters of the neural network. In general, two optional functions including thin-plate spline and Gaussian function are most often used on activated function between input and hidden layer [1]. While thin-plate spline function are most used in the time series modeling, Gaussian function is frequently used in pattern specification [1]. Finding parameters called weights based on the training at hand is called network training. In the supervised neural network, the training set is in the form of input-output pairs. The training algorithm will adapt the weights to fit the changing training set. After training, the RBF neural network can be used for pattern identification with the input data similar to the training set.

The project only focuses on the pattern classification implemented by supervised RBF neural network with Gaussian activated function. RBF neural network focused on pattern classification has been successfully applied into Iris flower recognition and number recognition [2]. For the Iris flower recognition, 3x8x4 RBF neural network is used to recognize three species of flower based on four features including length and width of sepal and petal. And for the number recognition, more complicated neural network is used to recognize numbers from “0” to “9”, injected with different level Gaussian errors. However, the recognition time will be greatly increased with the increasing complexity of the network. Additionally, complicated neural network will cause difficulty of logic elements’ placement and routing. So an idea is emerged. If we can build three simpler neural networks instead of a single complicated one, it will make the logic elements of the network more

flexible and easier for placement and routing. In my project, I introduce Triple Modular Redundancy technology [3] into the RBF neural network, which is expected to reduce the run time and increase the recognition rate but not increasing hardware overhead too much.

Focusing on the case of 7x8 pixel numbers recognition, I will introduce the basic structure of the RBF neural network in the first part of my paper. In the second part, I will discuss how to introduce the TMR techniques into the neural network. In the final part, I will compare the recognition rate, run time and hardware overhead of the single network with those of the network introduced by TMR.

RBF Network

Figure 1 shows the structure of the RBF network [2]. The input layers include 56 inputs denoted by vector $\boldsymbol{x}_i(i = 1,2,3\ldots56)$, indicating 7x8 pixel of each number. The neurons in the hidden layer implement the Gaussian activated function to extract the pixel features from input vector $\boldsymbol{x}_i$. Mathematically, the output of the hidden layer can be calculated in:

$$\boldsymbol{y}_i = exp(-\left|\left|\boldsymbol{x}_i - \boldsymbol{c}_{ij}\right|\right|^2/\sigma^2) \quad (1)$$

The vector $\boldsymbol{y}_i$ is the output of the jth neuron in the hidden layer $(j = 1,2,\ldots J)$. $\boldsymbol{c}_{ij}$ is the central point of the Gaussian function and σ is used to control the shape of $\boldsymbol{y}_i$. The output layer consists of 10 units denoted by $\boldsymbol{z}_k(k = 1,2,\ldots 10)$, indicating numbers from "0" to "9". The outputs of the overall network are computed through linear combination of $\boldsymbol{y}_i$ in:

$$\boldsymbol{z}_k = \boldsymbol{y}_i \cdot \boldsymbol{w}_{jk} + \boldsymbol{b}_k \quad (2)$$

Where $\boldsymbol{w}_{jk}$ is the weight matrix of the linear output layer and vector $\boldsymbol{b}_k$ is the bias. The values of $\boldsymbol{c}_{ij}, \sigma, \boldsymbol{w}_{jk}$ and $\boldsymbol{b}_k$ are four weights of the network and can be determined through training (figure 2). The recognition ability is various mainly based on the number of the neurons if given large enough training image.

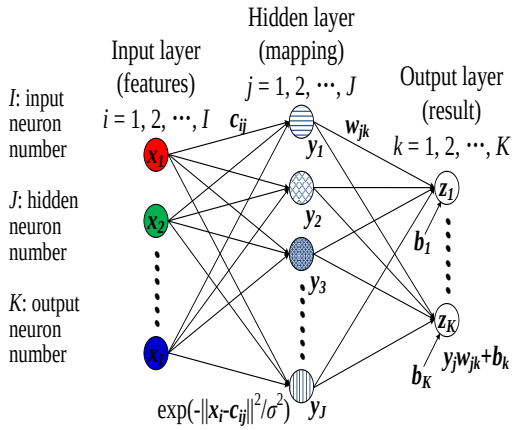

Figure 1 basic structure of RBF neural network

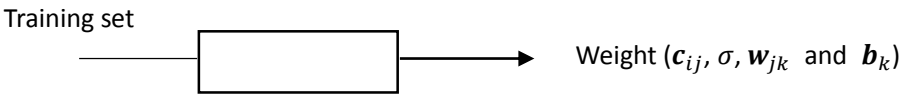

Figure 2 the training of the neural network

TMR System

TMR technique which is essentially to use two out of three voting concept has been applied to digital computer and many forms of redundancy to meet reliability requiments [3]. The goal of my paper is to introduce the TMR techniques to RBF neural network. As mentioned above, the ability of recognition of the network mainly depends on the number of neurons if given large enough trainning image. Figure 3 shows how to apply TMR techniques to conventional RBF neural network (called OMR). Divding the single neural network with n neurons into three sub-networks with n/3 neurons respectively and training the sub-network seperately to get three different weights. Given a recognition task, three sub-networks execute the task respectively and the final output is determined by two out of three after voting.

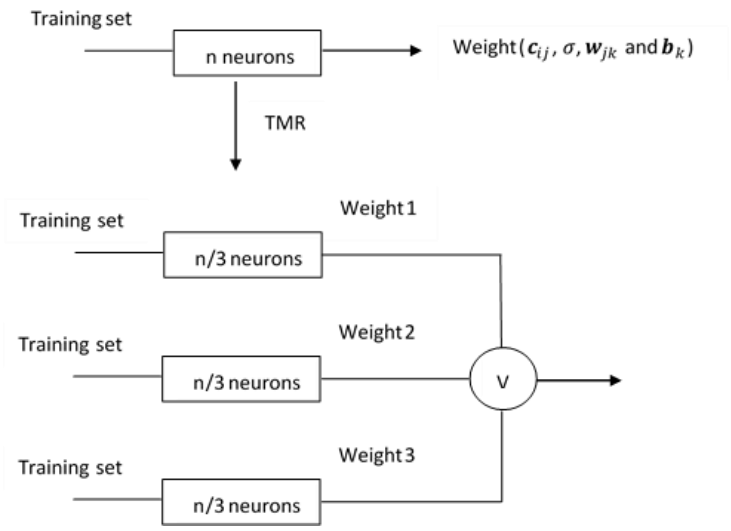

Figure 3 neural network in TMR

Comparison TMR with OMR

A. Performance (Recognition Rate)

Figure 4 shows the numbers injected with different Gaussian error levels (deviation from 0.1 to 0.4). In order to compare the abilities of recognition of the network in TMR and OMR, two networks in different schemes are built (see figure 3) to be tested. Figure 5 shows the recognition rate measured in the network based on TMR and OMR system. We can see that the performance of TMR has 1.0% to 2.0% improvement, compared with OMR.

Figure 4 number injected with different level Gaussian errors

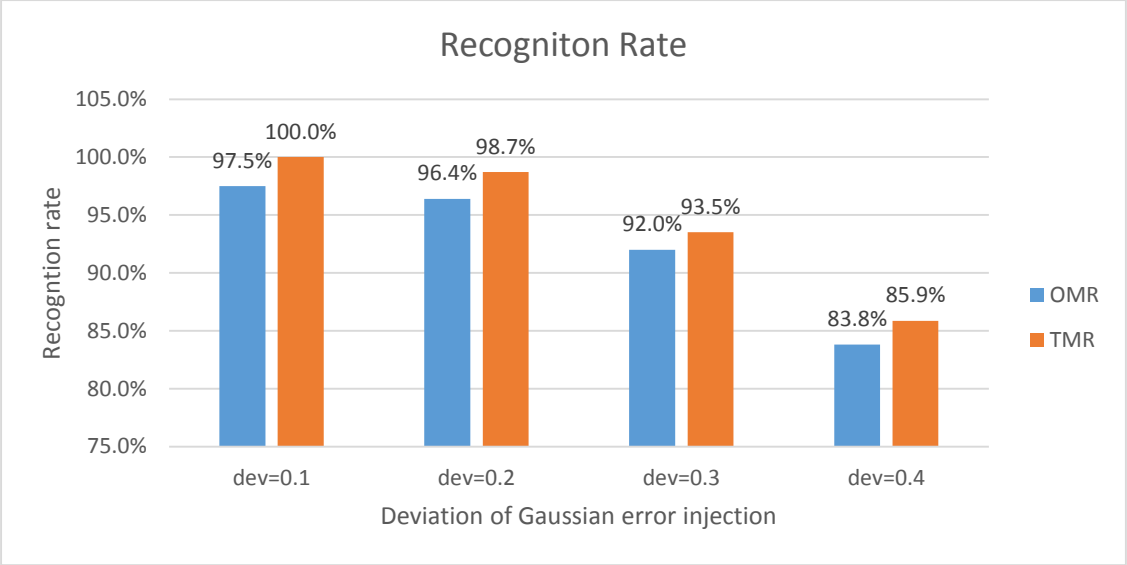

Figure 5 the recognition rate of the network in OMR and TMR

The mathematics quantification is used to verify the improvement of performance of the RBF neural network in TMR. Based on the basic knowledge of probability, the recognition rate of the network in TMR can be quantified:

$$P = r^3 + 3r^2(1-r) \quad (3)$$

where P is the recognition rate of the overall network in TMR and r is the recognition rate of the sub-network. Figure 6 shows the recognition rate with different neurons in different networks. In the graphs, the black lines describe the trend of recognition rates with increasing neurons in the network of OMR. While dashed line describes the trend of recognition rates with increasing neurons in the network of TMR based on the equation (3), the circle lines denote the trend in the network based on the actual experiment. According to Figure 6, we can see that the line derived in the equation (3) is very close to the line plotted based on the experiment. So the equation (3) is used to compute the overall recognition rate of the network in TMR is reasonable. However, the difference between the lines of TMR in theory and in experiment are mainly due to these two factors:

1. the recognition rate for each sub-network is not the same.
2. sub-networks are correlated with each other.

For the first factors, the equation (3) can be updated based on different recognition rates of each sub-networks:

$$P = r_1r_2r_3 + (1-r_1)r_2r_3 + r_1(1-r_2)r_3 + r_1r_2(1-r_3) \quad (4)$$

where $r_{1,}r_{2,}r_3$ are three recognition rates of three sub-networks.

For the second factors, two examples shown in Figure 7 are taken to verify if the recognition rate of the network in TMR is affected by the correlation between sub-networks. For Example 1, the recognition rate measured in the experiment P(exp) is very close to that P(theory) computed with equation (4). The correlation coefficient between each two sub-network approximately equals zero, which means each two sub-networks are uncorrelated. However, as shown in Example 2, the theoretical value 89.42% is reduced to 84.0% in the experiment that measured the recognition rate.

This reduction is because each two of three sub-networks are positively correlated as indicated by those correlation coefficients shown in the figure 7. So RBF neural network in TMR with three uncorrelated sub-networks can achieve high performance.

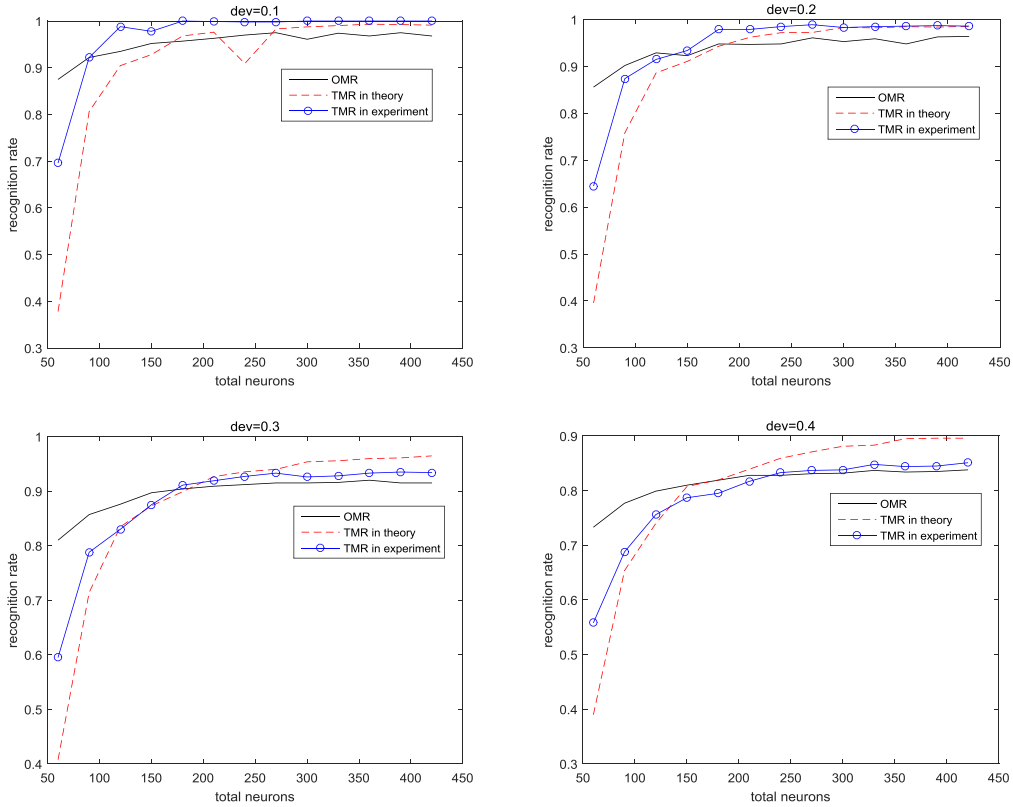

Figure 6 the recognition rate of different Gaussian error level injection in different schemes

Example 1: total neurons=300 and dev=0.1			
recognition rate		correlation coeffecient	
r1	98.10%	(r1,r2)	-0.0088
r2	99.60%	(r1,r3)	-0.0411
r3	92.00%	(r2,r3)	-0.0187
P(exp)	99.81%		
P(theory)	100%		
Example 2: total neurons=360 and dev=0.4			
recognition rate		correlation coeffecient	
r1	79.00%	(r1,r2)	0.3771
r2	77.30%	(r1,r3)	0.4856
r3	83.00%	(r2,r3)	0.3458
P(exp)	84.10%		
P(theory)	89.42%		

Figure 7 recognition rates based on different correlation coefficient

B. Run time

After applying TMR technique to RBF neural network, three sub-networks execute the recognition task in parallel. That is to say, while the run time measured is regarding to the network in OMR with n neurons, the run time measured is regarding to the network with n/3 neurons after introducing TMR technique. The time complexity based on the number of neurons is approved in mathematically in:

$$T = I \times J \times t_1 + J \times K \times t_2 = J \times (I \times t_1 + K \times t_2)$$

Where I is the number of the inputs, J is the number of the neurons, K is the number of the outputs, t_1 is the time complexity for each interconnection between input and hidden layer and t_2 is the time complexity for each interconnection between hidden and output layer. We can see that time complexity of the network T is linear with the number of neurons J. So the network in TMR can save run time twice, compared with the network in OMR with the same total neurons, which is matched with the Matlab simulation.

C. Hardware cost

The hardware cost I measured is through counting how many ALUs are used to do the operations and how much memory is needed to store the training weights in the task of recognizing the numbers injected with different level Gaussian errors based on the c++ code. Some equations are derived to compute the hardware cost in:

ALUs (adders, exponentials, multipliers, roots):

$$OMR = 477 \times neurons + 10$$
$$TMR = 477 \times neurons + 30$$

Memory($\boldsymbol{c}_{ij}, \sigma, \boldsymbol{w}_{jk}$ and $\boldsymbol{b}_k$):

$$OMR = 67 \times neurons + 10$$
$$TMR = 67 \times neurons + 30$$

Based on the equations above, we can see the cost of ALUs and Memory are very close in the same total number of neurons.

Conclusion

RBF neural network is set up to recognize the number from “0” to “9” injected with Gaussian errors of different deviation level. Two schemes, OMR (traditional scheme) and TMR (new scheme), are respectively applied into the network. Ensuring the approximate hardware overhead (same total neurons needed in the network), recognition rates and execution times are tested in two schemes with different error levels. And the results showed that the recognition rate and run time have improvements in TMR compared with OMR. In order to prove these improvements, the project also did the mathematical quantification to verify. So in the future, I hope to apply the TMR technology into different neural networks to see the performance and cost.

Reference

[1] Bors, Adrian. "Introduction of the Radial Basis Function (RBF) Networks."

[2] Ji, Yuan, and Feng Ran. "Using Stochastic Logics in Hardware Implementation of Radial Basis Function Neural Nework." *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, 2015, 880-83.

[3] Lyons, R. E. , Vanderkulk, W, "The Use of Triple-Modular Redundancy to Improve Computer Reliability," *IBM Journal of Research and Development*, 1962, Vol.6(2), pp.200-209.